Shock Wave

Cauz, Maxime; Albert, Julien; Wallemacq, Anne; Linden, Isabelle; Dumas, Bruno

Published in:
DocEng '21: Proceedings of the 21st ACM Symposium on Document Engineering

Publication date:
2021

Document Version
Peer reviewed version

Shock Wave: a Graph Layout Algorithm for Text Analyzing

Maxime Cauz*
Julien Albert*
maxime.cauz@unamur.be
julien.albert@unamur.be
Namur Digital Institute (NADI), University of Namur
Namur, Belgique

Anne Wallemacq
Isabelle Linden
Bruno Dumas
anne.wallemacq@unamur.be
isabelle.linden@unamur.be
bruno.dumas@unamur.be
Namur Digital Institute (NADI), University of Namur
Namur, Belgique

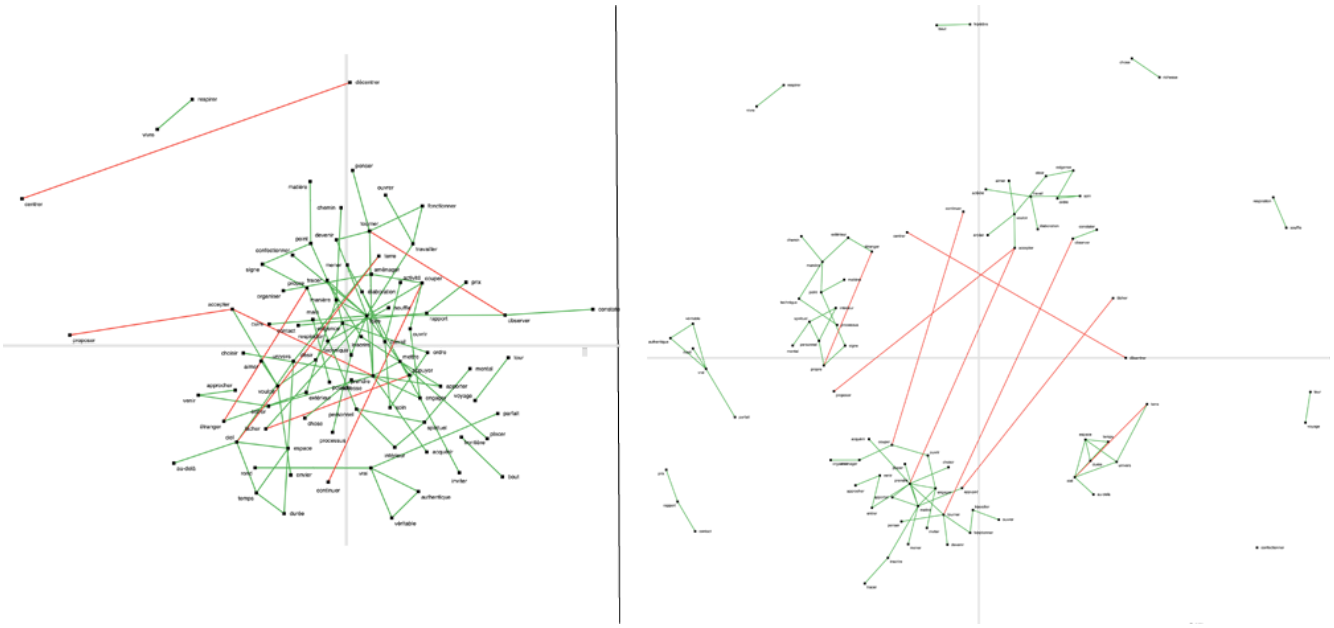

Figure 1: A graph layout in the *EVOQ* tool with on the left a random initial placement of nodes and on the right a initial placement done by the *Shock Wave*. As can be seen, our algorithm helps to distinguish the different semantic fields extracted by the user (i.e. groups of nodes linked by association relations association (green links)). And it highlights opposition relations (red links) which are a key part for the interpretation of these semantic fields.

ABSTRACT

The *EVOQ* tool offers researchers in social sciences a set of text analysis tools relying on the post-structuralist approach. This analysis approach relies on the identification of association and opposition relations between terms (words or expressions). The so-defined graph is presented in *EVOQ* by a node-link diagram. The *Shock Wave* is a placement algorithm specifically designed to be combined with a classical force-directed algorithm to produce a graph layout which meets the interpretability needs of the text analysts while preserving efficiency on large numbers of nodes. It structures the nodes on a circular placement with transversal opposition relations to highlight oppositions within the text concepts. Beyond our use case, the interest of *Shock Wave* lies in the fact that it is a novel method to present graphs of text with a strong emphasis on underlying semantic fields.

*Both authors contributed equally to this research.

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

CCS CONCEPTS

• **Human-centered computing** → **Graph drawings**; **Information visualization**; • **Applied computing** → **Document management and text processing**; *Law, social and behavioral sciences.*

KEYWORDS

Interactive Graph, Graph Visualization, Structural Analysis, Text Visualization, Relationships Visualization

ACM Reference Format:
Maxime Cauz, Julien Albert, Anne Wallemacq, Isabelle Linden, and Bruno Dumas. 2021. Shock Wave: a Graph Layout Algorithm for Text Analyzing. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Experts and researchers in social sciences and humanities frequently resort to manual or semi-manual text analysis techniques, which help them discover ambiguous, implicit or hidden aspects in articles or interviews. However, most of these techniques cannot easily be automated because they do not follow a generic algorithm but rather rely on the analysts' knowledge and -even- instinct. In this context, the EFFaTA-MeM (Evocative Framework For Text Analysis - Mediality Models) project which is a trans-disciplinary research project, explores new modes of interaction and visualization of texts (Linden et al. [8]).

As part of this project, the *EVOQ* tool, introduced in Clarinval et al. [1]. has been created as a text analysis tool for experts and researchers in social sciences and humanities. Its design follows a text analysis method inspired by post-structuralism [8] to support the actual practice of target users. These kinds of techniques encourage analysts to select words randomly or semi-randomly in the text and look for tensions between couples of words. As explained in Linden et al. [8], « the text is seen as a semantic landscape rather than a continuum extending from a beginning to an end ». In practice, the user proceeds by coupling words by association or opposition. For example [1], the user can associate words such as *white - pure - good* on one side and *black - dirty - bad* on the other to express two different semantic fields. Then, by adding a relation of opposition between *white* and *black*, he highlights the underlying ideas of *purity* and *impurity* and the tension structuring these semantic fields.

The *EVOQ* tool assists the user to define the relevant words or expressions (called *terms* in the application) of a text and to connect them by binary relations of association and opposition. In addition to selection assistance, the tool proposes a complementary set of idioms of visualization and interaction to facilitate the exploration of the meaning of a text (Clarinval et al. [1]).

The visualizations are: a) the text itself, b) a matrix of relations, c) a list of terms and a list of relations, d) a node-link diagram [9] where the terms and the relations are respectively the nodes and the links (red for opposition and green for association).

The node-link diagram is the main visualization idiom in *EVOQ*. It allows the user to place all the concepts of the text regardless of the reading order and to structure it according to the opposition relations (which are more important than association relations in the followed analysis method [8]). Beyond, the node-link diagram plays also the role of an interactive analysis space, where the user may manipulate and even play with the resulting visual representation. The user can add or remove terms or relations and, by moving nodes, may impact his interpretation of the text.

Therefore, the graph layout in the node-link diagram must meet some requirements which arise from the needs of our target users. It should respect our analysis method, allow the user to understand more easily the structure of a text, and reduce the number of nodes displacement by the user. In particular, the opposition relations should structure the graph layout because they are the most important according to the text analysis method. The nodes of a connected component by association relations, which form a semantic field, should also be placed near one other. And they should not overlap the area dedicated to another connected component (i.e. another semantic field). Finally, the general readability of the graph-layout should be preserved.

As stated in the literature over graph layout algorithms [4, 7] (i.e. algorithms that compute the position of nodes on a node-link diagram [9]), the main paradigm for computing layouts is to use force-directed methods. These methods consider the graph as « a physical system where nodes are attracted and repelled according to some force » [4]. To do so, they mainly use structural properties of graph. For instance, *Eades' Spring Embedder Algorithm* [2], which is one of the first main approaches and the basis for almost all force-directed techniques [4], considerates nodes as steel rings and links as springs. Starting from a random configuration, the nodes are positioned according to the resulting forces applied on them until the system reaches a stable state. However, force-directed methods alone can hardly deal with the strong user requirements of our case. The different types of links (i.e. associations and oppositions) and the prevalence of oppositions over associations need to clearly emerge from the structure of the resulting layout.

Numerous recent contributions in the domain focus on computing layouts for bigger graphs [7]. Despite addressing a different challenge from us, some techniques primarily designed to address this problem can also be relevant in our case. This is notably the case of multi-level techniques, which consist in combining sequentially different techniques to compute layout. These techniques are mainly motivated by the will to reduce the complexity of the global computation [4]. An example of these is proposed by Hendrickson and Leland [6] who used a multi-level technique to minimize the number of crossing links [4]. Another idea is to introduce a preprocessing step to initialize the algorithm rather than starting from a random state [7]. For example, Fowler and Kobourov [3] compare different methods of initial positioning with different force-directed algorithms to see if there is any aesthetic improvement in comparison to random positioning. This last idea is particularly relevant in our case because this preprocessing step may be an occasion to inject some user requirements into the graph layout. Inspired by this last idea, our solution to computing graph layouts, described in the two following sections, is to combine a force-directed algorithm and a placement algorithm. The force-directed algorithm follows the state of the art solutions. In contrast, *Shock Wave* is a novel placement algorithm specifically designed to answer our user requirements while preserving efficiency with bigger graphs.

2 FORCE-DIRECTED ALGORITHM

The force-directed algorithm used in *EVOQ* tool follows directly *Eades' Spring Embedder Algorithm* [2], but with some adaptations specific to the user requirements explained above. It works by

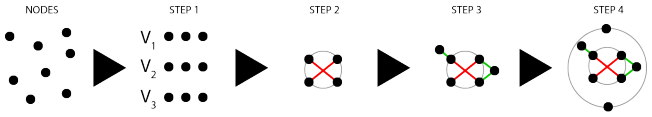

Figure 2: *Shock Wave* is divided in four steps.

applying different types of forces to nodes. First, a repulsive force is applied to all pair of nodes to ensure sufficient space between them. Second, an attractive force is applied to couples of nodes linked by an association relation. And third, a repulsive force is also applied to couples of nodes linked by an opposition relation. The strengths of these types of forces depend linearly on the distance between involved nodes. To ensure convergence, a limit to the number of iterations is also fixed (20 in our implementation).

All the repulsive/attractive forces performed are based on minimal and maximal distances for each type of relation or the absence of a relation. If two nodes have no relation between them, they must respect a minimal comfort distance to prevent overlap of labels (i.e. a node is represented by a dot with the term as label). The maximal distance between two associate nodes must be lesser than the minimal distance between two opposite nodes. Figure 1 is an example of the same graph in two different equilibrium state due to the different initial placement algorithm used.

3 SHOCK WAVE

The algorithm is divided in four successive steps (see Figure 2). The first step categorizes nodes in three sets V_1, V_2 and V_3 depending on their implication level in opposition relations (i.e. respectively direct, indirect or no implication). The three next steps place the nodes from the different sets on the visualization, in three waves starting from the center towards the border. The name *Shock Wave* comes from this arrangement in successive circles. The choice of circles as the main pattern to arrange nodes is motivated by the will to preserve the readability of the layout independently of the graph configuration. In addition to this, the first circle divides the space in two in order to keep the opposition relation inside and the rest outside. By this way, a circular arc never overlaps another node that the two attached to it (independently of the number of nodes on this circle) and the user can perceive more easily the different axes structuring the text. Finally, the circular construction guides the user's gaze towards the center where the opposition relations stand.

The nodes directly implied in an opposition relation are categorized in V_1. By positioning them, the algorithm places all the opposition relations. The nodes that have a path to a node of V_1 and are not in V_1 are included in V_2. The nodes in V_2 represent semantic fields that help enrich oppositions described in V_1. In the example of the introduction, not only *white* and *black* are in opposition but the concepts behind (i.e. *purity* and *impurity*), that may only be understood with the other associated words (i.e. *pure - good* and *dirty - bad*). Finally, the rest of the nodes forms the set V_3. This set is made up of terms that may be irrelevant to the analysis or terms that might be connected later to nodes from V_1 or V_2 by the user. Due to their yet unclear utility, these are placed on the

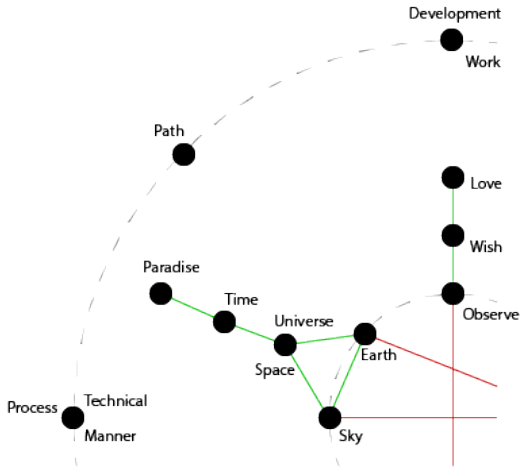

Figure 3: Disposition of nodes before the execution of the force-directed algorithm. The two circles are just displayed for the illustration.

outer layer of the graph. This way the user can clearly distinguish them and eventually decide what to do with them.

The placement of nodes in V_1 must highlight the groups of opposition relations by crossing them inside the first circle. Indeed, an associative relation may exist between nodes of V_1. Associative paths may also exist between two nodes of V_1 with some nodes of V_2. For example, it is the case in Figure 3 for *Sky* and *Earth*. Thus, the two opposition relations from these two terms are in the same group, while the opposition relation starting from *Observe* is in another. Thus, nodes linked by an associative path have to be closer to each other. So, the nodes of V_1 are grouped into *poles* in such a way that each pole contains all the nodes of V_1 that are connected by an association path (i.e. without an opposition relation, and by taking into consideration the transitive association through the nodes of V_2). Then, the poles are grouped into connected components by taking into account only the opposition relations (i.e. a connected component regroups all poles connected by transitive opposition relations). Next, in each connected component, the poles are distributed into two sets by minimizing the number of opposition relations between nodes of the same set. As the number of sets is limited to two, there can exist opposition relations between two poles of the same set (e.g. in the case of three poles in opposition to each other). Finally, these two sets are placed in opposition on the circle to create an axis. The sets of each connected component are intertwined to create the intersection between axes. In the Figure 3, the relation from *Observe* crosses the ones from *Earth* and *Sky* as two structural axes. The order of the nodes in each pole and the order of the poles themselves are given by minimizing the number of intersections between the relations. To reduce the number of permutations, the algorithm chooses the best order of nodes in each pole first, and then the best order of poles. For example, if we take two poles with respectively four and five nodes, the number of permutations reduces from 9! (362.880) with a brute force solution to 4! + 5! (144) with our heuristic. This trick respects the user requirements due to the fact that a pole represents a set of

associative nodes that must stay closer to one another. Once the order is known, the nodes of V_1 are placed on the inner circle where the radius is calculated to ensure that the minimal distance between two nodes is always the minimal distance for an opposition relation (cf. force-directed algorithm section). In other words, the nodes are the vertices of a regular polygon inscribed in the circle. Because the nodes of V_1 with an associative path between them are in the same pole, the force-directed algorithm can bring them closer without breaking the circle organization.

The third step consists in placing the nodes of V_2 depending on the nodes of V_1. They are placed outside the circle near their reference nodes in V_1 (a.k.a. nodes of V_1 linked by an associative path). The force-directed algorithm is in charge of computing the exact positions. Because the nodes of V_1 are placed according to the connected components, the reference nodes are close together on the circle. So the algorithm selects the one or two most centered to select the position of the nodes of V_2. To preserve user requirements, the nodes are placed on a line depending on the distance to the reference nodes (i.e. the size of the shorter path between them and the references nodes).

Finally, during the last step, the nodes of V_3 are placed on an outer circle around the nodes of V_1 and V_2. The nodes are grouped by connected components and separated by the force-directed algorithm like the nodes of V_2. Figure 3 illustrates the position of a few nodes of each category before executing the force-directed algorithm.

4 PRELIMINARY FEEDBACK

The initial evaluations of *Shock Wave* were done as part of the evaluation of *EVOQ*, with mainly a qualitative approach. The evaluation of *EVOQ* is a continuous process that is an integral part of the development of *EVOQ* and the validations we present below are part of this evaluation cycle. By this way, we are able to discover the practices and needs of users through observation, which facilitates their understanding and ultimately their integration into the tool.

Once the second prototype was mature enough, some sessions of evaluation, based on the quasi-empirical evaluation approach [5] were organized with three users who were not involved in the earlier development. These three users were all social scientists, with extensive expertise of manual and semi-automatic text analysis techniques. One of the three users had good knowledge of analysis techniques based on post-structuralism.

The first lessons learned with this preliminary evaluation are encouraging. Users underlined in particular the stimulating character of *EVOQ* and its ability to enrich the analysis process. They also enjoy that *EVOQ* supports their actual analysis practice rather replaces it like some other available tools do. On that regard, the different visualizations offered and in particular *Shock Wave* were considered as central tools supporting the analysis. Indeed, an important lesson from the evaluations is that researchers and analysts in social sciences and humanities can use some automation in the tools they use, as long as this automation supports their detailed, word-per-word analysis instead of trying to replace it. This evaluation phase is still in progress, with the intention of expanding the panel of users.

5 CONCLUSION AND FUTURE WORK

As stated in the introduction, our challenge was to provide a graph layout computation method for the node-link diagram in *EVOQ* tool. This method must meet a set of specific user requirements about the produced layout while preserving efficiency with larger graphs. The combination of a force-directed algorithm and *Shock Wave* provides an efficient solution by respecting the user requirements and the application constraints. First, the placement algorithm places the nodes to highlight the structural axes represented by the opposition relations. Then, the structure of the computed layout does not change even if the user adds or removes many nodes. Finally, the algorithm uses some heuristics and the force-directed algorithm to reduce drastically the execution time. A more formal evaluation must be carried out and is in preparation to confirm the interest of *Shock Wave*. However, preliminary feedback from users confirm the interest of our approach.

Nevertheless, some improvements will be explored in future work. It could be interesting to include multi-level graphs to abstract the concepts behind the associative connected components. Another improvement concerns the distribution in two sets of poles before placing them on the circle. The number of sets could then be calculated based on a larger cycle of oppositions between poles.

Finally, *Shock Wave* opens new perspectives in graph layout with different types of relations. Indeed, the starting point was to design a graph layout method that takes into account a 2-level hierarchy of relations (association and opposition). So, a natural development would be to make our method able to deal with more complex hierarchies of relations (e.g., by adding strong and weak opposition relations). Another idea would be to extend our method to graphs without a hierarchy between the types of relations.

REFERENCES

[1] Antoine Clarinval, Isabelle Linden, Anne Wallemacq, and Bruno Dumas. 2018. Evoq: A Visualization Tool to Support Structural Analysis of Text Documents. In *Proceedings of the ACM Symposium on Document Engineering 2018* (Halifax, NS, Canada) *(DocEng '18)*. Association for Computing Machinery, New York, NY, USA, Article 27, 10 pages.

[2] Peter Eades. 1984. A heuristic for graph drawing. *Congressus numerantium* 42 (1984), 149–160.

[3] J. Joseph Fowler and Stephen G. Kobourov. 2012. Planar Preprocessing for Spring Embedders. In *Proceedings of the 20th international conference on Graph Drawing*. 388–399.

[4] Helen Gibson, Joe Faith, and Paul Vickers. 2013. A survey of two-dimensional graph layout techniques for information visualisation. *Information Visualization* 12, 3-4 (jul 2013), 324–357.

[5] Rex Hartson and Partha S. Pyla. 2012. Chapter 13 - Rapid Evaluation Methods. In *The UX Book*, Rex Hartson and Partha S. Pyla (Eds.). Morgan Kaufmann, Boston, 467–501.

[6] Bruce Hendrickson and Robert Leland. 1995. A multilevel algorithm for partitioning graphs. In *Proceedings of the 1995 ACM/IEEE conference on Supercomputing (CDROM) - Supercomputing '95*. ACM Press, New York, New York, USA, 28–es.

[7] Stephen G. Kobourov. 2013. Force-Directed Drawing Algorithms. In *Handbook of Graph Drawing and Visualization*, Roberto Tamassia (Ed.). Chapman and Hall/CRC, 398–423.

[8] Isabelle Linden, Anne Wallemacq, Bruno Dumas, Guy Deville, Antoine Clarinval, and Maxime Cauz. 2020. Text as Semantic Fields: Integration of an Enriched Language Conception in the Text Analysis Tool Evoq®. In *Research Challenges in Information Science*, Fabiano Dalpiaz, Jelena Zdravkovic, and Pericles Loucopoulos (Eds.). Springer International Publishing, Cham, 543–548.

[9] Raga'ad M. Tarawneh, Patric Keller, and Achim Ebert. 2012. A general introduction to graph visualization techniques. *OpenAccess Series in Informatics* 27 (2012), 151–164.